An Incremental Clustering of Gene Expression data

Rosy Das* Dhruba K. Bhattacharyya* Jugal K. Kalita$^{\alpha}$
*Dept. of Computer Science and Engineering$^{\beta}$
Tezpur University, Tezpur, India
Email: {rosy8, dkb}@tezu.ernet.in
$^{\alpha}$Dept. of Computer Science
University of Colorado, Colorado Springs, USA
Email: kalita@eas.uccs.edu

***Abstract*—This paper presents an incremental clustering algorithm based on DGC, a density-based algorithm we developed earlier [1]. We experimented with real-life datasets and both methods perform satisfactorily. The methods have been compared with some well-known clustering algorithms and they perform well in terms of z-score cluster validity measure.**

***Index Terms*—Gene expression, dissimilarity measure, clustering, density based, incremental clustering.**

I. Introduction

Data mining techniques are useful in understanding gene function, gene regulation, cellular processes and subtypes of cells. According to [2], most data mining algorithms developed for gene expression deal with the problem of clustering. Microarrays have made it possible to observe the expression levels of thousands of genes simultaneously across various conditions or processes. Clustering algorithms group similar genes into the same cluster based on the similarity among their expression profiles. With the growth of microarray technology, more and more data are being generated and hence new and efficient clustering methods must continue to be developed to handle the exponential growth of biological data.

A large number of clustering techniques have been been reported for analyzing gene expression data [3]. The current information explosion, fuelled by the availability of the World Wide Web and the huge amount of microarray experiments being conducted, has led to the ever-increasing volume of data. Therefore, there is a need to introduce incremental clustering so that updates can be clustered in an incremental manner. In [4], the authors present an incremental clustering approach based on the DBSCAN [5] algorithm. In [6], an efficient method for modifying a set of association rules has been presented. In [7], an incremental clustering algorithm for information retrieval applications is presented. A one pass clustering algorithm for relational datasets is proposed in [8]. Rough set theory was employed in the incremental approach for clustering interval datasets in [9]. Though a lot of research has been done on incremental clustering for other application domains, incremental clustering in gene expression data has not been exploited much. In [10], an incremental genetic K-means algorithm is presented. In [11], an incremental gene selection algorithm using a wrapper-based method is presented which reduces the search space complexity since it works on the ranking directly.

In this paper, we introduce an incremental density based clustering technique (*incDGC*) for gene expression data, which is designed based on our existing density based clustering technique DGC [1]. Next, we introduce our DGC based on which the proposed incremental density-based clustering is developed.

II. DenGeneClus (DGC)

DGC [1] works in two phases discussed in detail below.

A. Phase I: Normalization and Discretization

This phase is a two step process. The first step deals with normalization of the gene expression data to have mean 0 and standard deviation 1. Low variance data as well as data having more than 3-fold variation are filtered out in this step. The second step of discretization is then performed on this normalized expression data where the regulation pattern, i.e. up- or down- regulation in each of the conditions for a particular gene plays an important role. Suppose, G is the set of all genes and T is the set of all conditions. Let $g_i \in G$ be the i^{th} gene and $t_j \in T$ be the j^{th} condition. The expression value of gene g_i at condition t_j is given by $\hbar_{i,j}$. The discretization step gives us the regulation pattern of genes across conditions. For a particular gene, the regulation pattern is computed (for all conditions except the first) based on the previous condition value. For the first condition, t_1, the discretized value of gene g_i is directly based on $\hbar_{i,1}$. For the ${t_{j+1}}^{th}$ condition, the discretized value is computed w.r.t. the ${t_j}^{th}$ condition, i.e., it is based on $\hbar_{i,j+1}$ and $\hbar_{i,j}$. While discretizing, following two cases occur:

Case 1: For the first condition t_1. The discretized value of gene g_i at t_1.

$$\xi_{i,1} = \begin{cases} 1 & \text{if } \hbar_{i,1} > 0 \\ 0 & \text{if } \hbar_{i,1} = 0 \\ 2 & \text{if } \hbar_{i,1} < 0 \end{cases}$$

Case 2: For the remaining conditions $(T - \{t_1\})$. The discretized value of gene g_i at t_{j+1}

$$\xi_{i,j+1} = \begin{cases} 1 & \text{if } \hbar_{i,j} < \hbar_{i,j+1} \\ 0 & \text{if } \hbar_{i,j} = \hbar_{i,j+1} \\ 2 & \text{if } \hbar_{i,j} > \hbar_{i,j+1} \end{cases}$$

where $\xi_{i,j+1}$ is the discretized value of gene g_i at condition t_{j+1} (j = 1,..T). Each gene now has a regulation pattern ($\wp$)

978-1-4244-5612-3/09/$26.00

of 0, 1, and 2 across the conditions or time points.
Once $\wp$ of each gene is obtained, the second phase i.e. the clustering process is initiated.

B. Phase II: Density Clustering of Genes

The clustering of genes is initiated with the finding of the maximal matching genes with respect to the regulation pattern. The basic idea behind detecting a cluster is that within each cluster we have a typical density of genes having similar expression patterns which is considerably higher than that outside the cluster. Furthermore, the density within the areas of noise is lower than the density in any of the clusters. In the following, we try to formalize this intuitive notion of clusters and noise in a database G of genes. The key idea is that for each gene of a cluster, the neighborhood must contain at least δ genes which have similar expression pattern (regPattern). The shape of a neighborhood is determined by the choice of a distance function for two genes g_i and g_j, denoted by $D(g_i, g_j)$. Note, that our approach works with any distance measure and hence there is provision for selecting the appropriate similarity function for some given application. In this paper we use our own dissimilarity measure [12] which has been discussed in brief in the previous section.

1) Basis of the Clustering Approach: Regulation matching, order preservation and proximity are the three pillars based on which the clustering technique (DenGeneClus or DGC) is designed.

- Regulation Matching: For a particular gene g_i, the maximal matching regulation pattern is found. All genes having the same maximal matching regulation pattern w.r.t. g_i are grouped into the same cluster.
- Order Preservation: We follow the order preservation requirement in a set [13] in the following way. For a condition set $t \subset T$ and a gene $g_i \in G$, t can be ordered in a way so that the expression values are ordered in ascending order. By order ranking, we search for the expression levels of genes within a cluster which induce ordering of the experiments (conditions). Such a pattern might arise, for example, if the experiments in t represent distinct stages in the progress of a disease or in a cellular process and the expression levels of all genes in a cluster vary across the stages in the same way [13].
  Each gene has a rank which gives the permutation order of that gene across conditions t. The rank is calculated according to the expression values of a gene across conditions or in other words, the elements of the rank pattern are given in ascending order of their expression values.
- Proximity: The proximity between any two genes g_i and g_j is given by $D(g_i, g_j)$ where D is any proximity measure like Euclidean distance, Pearsons Correlation etc.

The following definitions and lemmas provide the theoretical basis of the proposed incDGC approach. Some definitions are given based on the notion of density available in [5].

Definition 1: Match: Let $\wp_{g_i}$ and $\wp_{g_j}$ be the regulation pattern of two genes g_i and g_j. Then, the match (M) between g_i and g_j is given by the number of agreements, $Num\_Agreements$ (i.e., the number of condition-wise common regulation values excluding condition 1) between the two regulation patterns: $M(g_i, g_j) = Num\_Agreements(\wp_{g_i}, \wp_{g_j})$

Definition 2: Maximal Match: Gene g_i is referred to as maximally matched (MM) with gene g_j if $\forall g_j \in G - \{g_i\}$, $max\{M(g_i, \wp_{g_j}) \geq \delta\}$.

Definition 3: Maximal Matching Regulation Pattern: If a gene g_i maximally matches gene g_j, say, then the pattern $\wp'_{g_i}$ and $\wp'_{g_j}$ formed by taking the subset of conditions where both $\wp_{g_i}$ and $\wp_{g_j}$ match is referred to as the Maximal Matching Regulation Pattern (MMRP), i.e., the longest matching subsequence of the regulation pattern.
MMRP of genes g_i and g_j is computed as follows.

$$\wp'_{g_i} = \wp'_{g_j} = \begin{cases} 1 & \text{if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = 1 \\ 0 & \text{if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = 0 \\ 2 & \text{if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = 2 \\ \times & \text{otherwise} \end{cases}$$

Here t refers to the subset of conditions of maximum cardinality (T-1) ($i.e. t = 2, 3, \cdots, T$).
Each gene has a rank which gives the permutation order of that gene across conditions $t \subset T$. The rank is calculated according to the expression values of a gene across conditions *i.e.,* the elements of the rank pattern are given by their ranking in ascending order of their expression values. The rank of a gene is calculated as follows:

1) For a gene g_i, find $\wp'_{g_i}$
2) Rank g_i in ascending order according to the expression values where $\wp'_{g_i,t} \neq \times$;

Definition 4: θ-neighborhood: The θ-neighborhood of a gene g_i, denoted by $N_\theta(g_i)$ is defined by,
$N_\theta(g_i) = \{g_i \in G \mid \forall g_j \in G - \{g_i\}, D(g_i, g_j) \leq \theta\}$
where, D may be any distance measure such as Euclidean, Pearson's correlation, or our dissimilarity measure [12].

Definition 5: Core Gene : A gene g_i is said to be a Core gene w.r.t. θ if

i. $g_j \in N_\theta(g_i)$, where $g_j \in G$
ii. $\mid N_\theta(g_i) \mid \geq \sigma$
iii. $\wp'_{g_i} = \wp'_{g_j}$
iv. $Rank(g_i) = Rank(g_j)$

where σ is a user defined threshold for the minimum number of genes in the θ-neighborhood of g_i.

Definition 6: Directly Reachable Gene: A gene g_i is directly reachable from gene g_j w.r.t. θ if g_j is a core gene and $g_i \in N_\theta(g_j)$.
Direct reachability relation of a gene is symmetric for pairs of core genes. However, in case of a pair of core and non-core genes, it may not be valid.

Definition 7: Reachable Gene: A gene p is said to be reachable from gene q w.r.t. θ if there is a chain of genes $P_1, P_2, \cdots, P_n$, where $P_1 = q$, $P_n = p$ such that P_{i+1} is directly reachable from P_i.

Thus, reachability relation is a canonical extension of direct reachability [5]. This relation is transitive, but is not symmetric. However, over the gene expression domain, reachability is symmetric for core genes.

Definition 8: Connected Genes: A gene g_i is said to be connected to another gene g_j if both g_i and g_j are reachable from another gene g_k w.r.t. θ.
Connectivity is a symmetric relation. For reachable genes, the relation of connectivity is also reflexive.

Definition 9: Cluster: A cluster C w.r.t. θ is a non-empty subset of G and $\mid C \mid \geq \sigma$ satisfying the following conditions:

i. $\forall g_i, g_j$ if $g_i \in C$ and g_j is reachable from g_i w.r.t. θ then, $g_j \in C$(reachability).
ii. $\forall g_i, g_j \in C : g_i$ is connected to g_j w.r.t. θ (connectivity)

Therefore, a cluster can be defined as a set of reachable and/or connected genes.

Definition 10: Noise: Let C be the set of clusters of the dataset G w.r.t. parameter θ. Noise is defined as the set of genes not belonging to any cluster $C_i \in C$. In other words,

$$noise = \{g_i \in G \mid \forall i : g_i \notin \{C_1 \cup C_2 \cup \cdots \cup C_n\}\}$$

Also, a gene g_i is said to be a noise gene if it does not satisfy the θ-neighborhood condition i.e., $\mid N_\theta(g_i) \mid = \phi$
Note that any cluster C_i w.r.t. θ contains at least two genes (i.e. $\sigma = 2$) to satisfy the core gene condition.

2) The Clustering Process: DGC: The intuitive notion behind DGC is that given the parameter θ we can discover a cluster in a two-step approach. First, an arbitrary gene is chosen as the seed which satisfies the core gene condition. Second, all genes reachable from the seed are retrieved. These two steps result in a cluster containing the seed.

Cluster identification starts with an arbitrary gene and finds the MMRP with other unclassified genes. For regulation pattern matching, two genes are matched w.r.t. regulation across the conditions starting from condition 2. Condition 1 is not considered as its regulation is w.r.t. the expression level rather than the previous condition. If the arbitrary gene is a core gene, cluster expansion proceeds with this core gene by finding reachable and connected genes from this core gene. All reachable and connected genes in a particular iteration of the clustering process are grouped into the same cluster. The process then recursively continues until all genes are classified. This expansion process can be summarized in terms of the following steps:

i. Start with an arbitrary unclassified gene g_i and find its rank order and regulation pattern.
ii. Call $get\_Core(g_i)$
iii. For each core gene g_i
    a) Find all reachable and connected genes w.r.t. g_i
    b) Classify all those genes with the same Cluster-id
iv. Increment i
v. Repeat steps ii. to iv. until no more core gene is found
vi. Repeat steps i. to v. until all genes are classified.

Here, $get\_Core(g_i)$ returns a true value if gene g_i is core. Clustering result of DGC using our dissimilarity measure is reported in Section IV-A.

III. INCDGC: INCREMENTAL DGC

DGC as discussed in the previous section can be used for static gene expression data. Due to the huge amount of microarray experiments, whenever new gene expression data become available, it is highly desirable to perform the updates (the clustering) of these newly arrived genes incrementally. Therefore, we propose an incremental clustering algorithm, incDGC, based on DGC. The intuitive idea behind incDGC is based on [4]. Due to the density based nature of DGC, the insertion of a gene affects the current clustering only in the neighborhood of this gene. It has been found that the incremental algorithm yields the same result as DGC. It would be a significant achievement if we could update the clustering obtained by DGC (on the old database) to handle the new updates. We examine the parts of an existing clustering effort affected by an update and present the algorithm incDGC, for incremental updates of a clustering after insertions.

The changes in the clustering of the gene database D_G are restricted to the neighborhood, i.e., $N_\theta(g_i)$, of an inserted gene g_i. The previous core genes retain their core property but, non-core genes (border genes or noise genes) in $N_\theta(g_i)$ may become core. Thus, new density connections may surface, i.e., chains $g_1, \cdots, g_n$, $g_1 = r, g_n = s$ with g_{j+1} directly density reachable from g_j for two genes r and s may arise which were not density reachable before the insertion of g_i. Thus, one g_j for $j < n$ must be contained in $N_\theta(g_i)$. Figure 1(a) shows an example database of genes illustrated in 2D and gene g_i is to be inserted. Each of the points represents a gene. The genes a and b are density connected w.r.t. θ and $\sigma = 4$ without using any gene $\in N_\theta(g_i)$. On the other hand, genes r and s are density connected via $t \in N_\theta(g_i)$ if the gene g_i is present and gene g_i is directly density reachable from t, i.e., gene t and g_i have same regulation also. Thus, the cluster membership of r and s is dependent on the presence or absence of g_i.

The insertion of a gene g_i may result in a change of cluster membership of genes in θ-neighborhood of g_i and all genes density reachable from one of these genes in $D_G \cup g_i$. While inserting g_i the following cases may occur:

1) **Fusion:** If g_i is reachable from exactly one cluster C_i, then g_i and possibly some noise genes are fused into cluster C_i.
2) **Creation:** g_i may become core w.r.t. some other noise or unclassified gene(s) and may lead to the formation of a new cluster.
3) **Merge:** Gene $g_k \in N_\theta(g_i)$ and g_k becomes core after insertion of g_i. Also, gene g_r is core and $g_r \in N_\theta(g_k)$, if all g_k, g_r belong to different clusters, all these clusters as well as g_i are merged to form one cluster.
4) **Noise:** g_i is neither a core gene nor it is density reachable from any other core gene. Moreover, insertion of g_i does not produce any new core genes. Then g_i is noise gene and no density-connections are changed.

The above four cases are depicted in Fig. 1(b) for 2D illustration where $\sigma = 4$. The incDGC starts with a newly

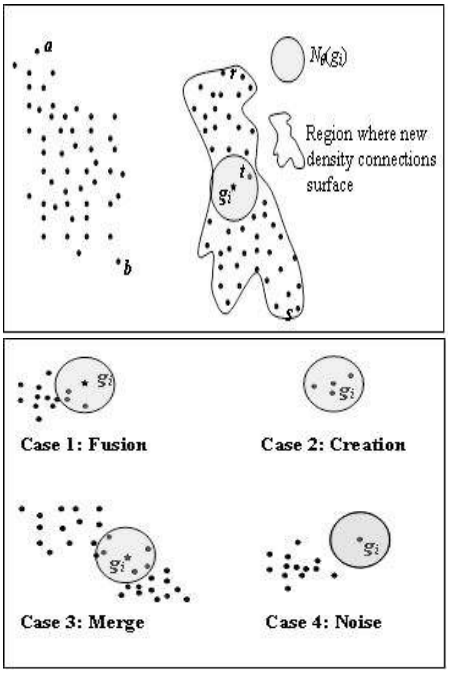

Fig. 1. (a) Example dataset of genes and (b) The different cases of insertion

inserted gene g_i and finds its regulation pattern. Each cluster formed on the old database will have a cluster MMRP and there can be the following cases:
i) g_i is matched with each of the cluster MMRPs: If it matches with exactly one cluster C_i, incDGC proceeds with the gene g_i (and can be a viable case either for case 1, 2 or 4 above). Only the genes in the θ-neighborhood of g_i which belong to C_i or are unclassified become the seeds for cluster expansion.
ii) If g_i matches more than one cluster, the seeds for cluster expansion are those genes belonging to these clusters as well as the unclassified genes belonging to θ-neighborhood of g_i (Case 3).
iii) If g_i matches none of the clusters, either case 2 or 4 might occur.

For a gene expression database of n genes and y inserted genes the following theorems and lemmas are derived.

Theorem 1: incDGC has performance $O(n+y)$ in the worst case.

Proof: Assume m clusters have been detected by DGC on the database D_G of size n. For an insertion of y genes, the cardinality of the updated database D_{upd} becomes $(n+y)$. For finding matching profile(s), incDGC compares with m profiles where $m << n$. This results in a complexity of $O(m)$. Once matching profile(s) are identified, neighborhood processing starts. Suppose x is the average number of genes in a cluster. Let $g_i \in \{D_{upd} - D_G\}$ be an inserted gene and g_i matches with k clusters $(k = 1, 2, \cdots, m)$. Then the neighborhood query searches $((x \times k) + z)$ genes where $(x \times k) << n$ and z is the set of unclassified genes $\in D_{upd}$. This will give a complexity of $O((x \times k) + z)$. Once the neighborhood of g_i is identified, the four cases discussed above are checked. Out of the four, the merging case can be found to be more costly which can be at most $O(x \times k)$.Therefore,
$Total\ time\ complexity = O(m) + O((x \times k) + z) + O(x \times k)$
$= O((x \times k) + z)$
In the worst case, $k = m$, $Total\ time\ complexity = O((x \times m) + z) \approx O(n + y)$ ■

Observation 1: Clustering result obtained by INCDGC is the same as the clustering result obtained by DGC

Lemma 1: Let g_i be an inserted gene and genes $g_x \in C_1$ and $g_y \in C_2$, where C_1, C_2 are two clusters. If g_i becomes core and both g_x and g_y are reachable from g_i, then C_1 and C_2 are merged.

Proof: Suppose $g_x \in C_1$ and $g_y \in C_2$ and inserted gene g_i is found to be a core gene. Also, let g_x and g_y be reachable from g_i. Then g_x is density connected to g_y and as per Definition 9, $g_x$and g_y belong to the same cluster i.e., clusters C_1 and C_2 should be merged and hence the proof. ■

Lemma 2: Let g_i be an inserted gene and genes $g_x \in C_1$ and $g_y \in C_2$, where C_1, C_2 are two clusters. If g_i is not core and g_i is reachable from both g_x and g_y, then, $g_i \in C_1$ or $g_i \in C_2$.

Proof: Assume, g_i be an inserted gene and g_i is not core. Also, let g_i be reachable from both the clusters $g_x \in C_1$ and $g_y \in C_2$ then according to Definition 9, $g_i \in C_1$ and $g_i \in C_2$. However, as per *lemma* 1, C_1, C_2 cannot be merged as g_i is not core. Therefore g_i can be included in any of C_1 or C_2 and hence the proof. ■

A significant advantage of incDGC is that genes in the $N_\theta(g_i)$ having MMRP different from that of g_i are not considered for cluster expansion. This in turn reduces the computational cost of the algorithm significantly.

IV. Performance Evaluation

In this section, we evaluate the efficiency of incDGC versus DGC. The methods were implemented in Java in Windows environment. The methods was evaluated with three real-life datasets. *Dataset 1* was taken from [14]; it contains expression of 6089 genes of *Saccharomyces cerevisiae*. Expression levels were measured at seven time points during the diauxic shift. *Dataset 2* [15] contains the expression data of *Saccharomyces cerevisiae* for 6218 genes at 17 time points. *Dataset 3* [16] contains the rat Central Nervous System (CNS) data for gene expression patterns of 112 genes measured at nine different developmental time points. However, due to space constraint only the results of *Dataset 1* are reported. The detailed results can be obtained from http://202.141.129.18/~rosy8.

A. Results: DGC

We exhaustively tested DGC on the above datasets with $\sigma = 2$. The value of σ was taken to be 2 as we went for an exhaustive search for different patterns. We have used the Euclidean distance and our dissimilarity measure for D and the value of $\theta = 2$. The values of θ and σ was obtained by experimental fine tuning. On experimentation with various real-life and synthetic datasets, the method was found to give satisfactory results. We compared our algorithm with that of

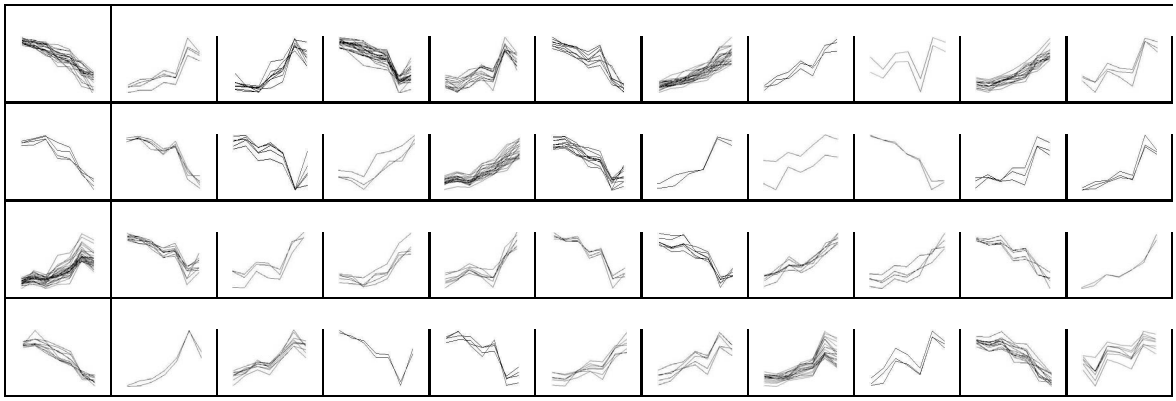

Fig. 2. Result of DGC on the reduced form of Dataset 1 using our dissimilarity measure

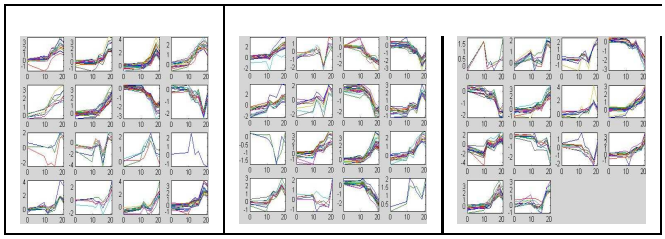

Fig. 3. Result of K-means on the reduced form Dataset 1 at cutoff = 46

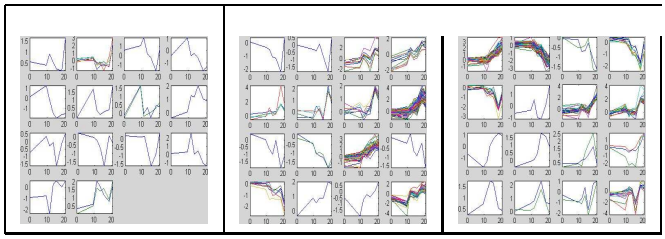

Fig. 4. Result of UPGMA on the reduced form Dataset 1 at cutoff = 46

the K-means [17] and hierarchical clustering (UPGMA) [18] algorithms. The results obtained by our method over a reduced form of Dataset 1 are shown in Fig. 2. The dataset was reduced by filtering out the low variance and low entropy genes from the data. We note here that the clusters obtained by our algorithm are detected automatically and unlike K-means no input parameter for number of clusters is needed. We have tested K-means with $k = 16, 20, 30, 40, 48$. Since our method gave a total of 47 clusters (when Euclidean distance was used) and 44 clusters (when our dissimilarity measure was used) for the reduced form of Dataset 1, we also tested k-means algorithm for k = 44 and 47 respectively. Similarly, UPGMA algorithm was tested for cutoff = 43, 44, 47 and also for various other values. In Fig. 3, the clusters generated by K-means on the reduced form of Dataset 1 are given. In Fig. 4, clusters generated from the reduced form of Datset 1 using UPGMA at cutoff= 46 and 176 are shown respectively. Finally, to validate the cluster results, the cluster validity measure z-score was used and the results were compared with the different clustering algorithms.

B. Cluster Quality

To assess the quality of DGC, we need an objective external criterion. A statistical rating of the relative gene-expression activity in each cluster and GO term has been done. In order to validate our clustering result, we employ z-score [19] as the measure of agreement. A higher value of z indicates that genes would be better clustered by function, indicating a more biologically relevant clustering result. We have used Gibbons ClusterJudge [19] tool to calculate the z-score. To test the performance of the clustering algorithm, we compare the clusters identified by our method with the ground truth and with the results from K-means and UPGMA. In this paper, the reported z-score is averaged over 50 repeated experiments. The result of applying the z-score on the reduced form of Dataset 1 is shown in Table I. In this table DGC was compared with the well known K-means and the agglomerative hierarchical algorithm, UPGMA. Table 1 clearly shows that our method outperforms both k-means and UPGMA w.r.t. the cluster quality. The z-score values obtained from clustering the full Dataset 1 is given in Table II. As can be seen in the table, our method performs better than K-means and hierarchical clustering. We note here that unlike K-means our method does not require the number of clusters as an input parameter. It detects the clusters present in the dataset automatically and gives the rest as noise. Also, UPGMA requires the parameter cutoff as input to the algorithm. From all the three tables it can be seen that the cluster result gives better clustering at $\theta = 2$ for Dataset 1. The z-score for DGC and incDGC was also found and is shown in Table I and Table II. It can be seen from the tables that incDGC discovers all the clusters as DGC as is depicted by the z-score.

C. Execution Time Performance

The execution times of DGC and incDGC was compared by increasing the size of the dataset with updates of 500 genes for each iteration. The execution time performance of both the algorithms is illustrated in figure 5. From the graph, it can be

TABLE I
Z-SCORES FOR DGC, K-MEANS AT K=16 AND 46 AND UPGMA USING AVERAGE LINKAGE AT CUTOFF = 16 AND 46 FOR THE REDUCED FORM OF DATASET 1

Method Applied	No. of Clusters	z-score	Total no. of genes
UPGMA	16	0.285	614
k-means	16	-0.366	614
UPGMA	46	1.69	614
k-means	46	0.193	614
DGC at $\theta = 0.7$	46	5.38	614
DGC at $\theta = 1$	44	6.55	614
DGC at $\theta = 1.5$	44	6.41	614
DGC at $\theta = 2$	44	7.07	614
DGC at $\theta = 2.7$	44	6.58	614
incDGC	**46**	**7.07**	**614**

TABLE II
Z-SCORES FOR DGC, AND UPGMA USING AVERAGE LINKAGE AT CUTOFF = 176 FOR THE FULL DATASET 1

Method Applied	No. of Clusters	z-score	Total no. of genes
UPGMA	176	9.7	6089
k-means	176	NA	6089
DGC at $\theta = 0.7$	176	9.12	6089
DGC at $\theta = 1$	128	7.02	6089
DGC at $\theta = 1.5$	120	11.2	6089
DGC at $\theta = 2$	118	12	6089
DGC at $\theta = 2.7$	120	11.2	6089
incDGC	**176**	**12**	**6089**

seen that with increase in the size of the updated database, the performance of DGC degrades unlike incDGC.

V. CONCLUSION

This paper presents an incremental clustering algorithm (incDGC) based on DGC [1]. DGC does not require the number of clusters apriori and the clusters obtained by DGC have been found satisfactory on visual inspection and also based on z-score for three real datasets. The regulation based cluster expansion overcomes the problem of maintaining the pattern information usually linked with the different clustering approaches due to traditional similarity measures. The incDGC algorithm brings down the cost of performing DGC on the whole database after insertions are carried out. The number of neighborhood queries are scaled down much effectively than DGC if allowed to run on the whole updated data. Moreover, the incDGC always gives the same result as a DGC run on the whole database and is also much faster than DGC.

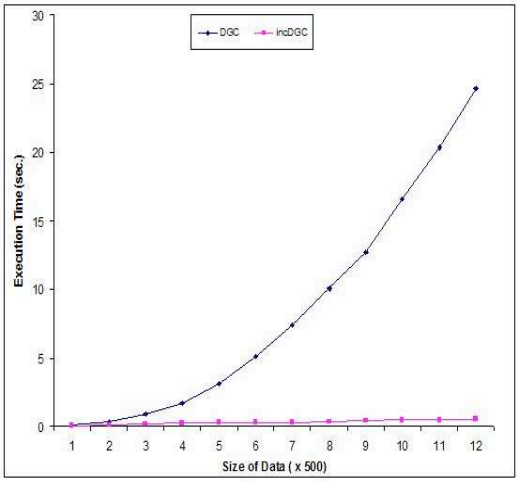

Fig. 5. Execution Times of DGC and incDGC with increase in the size of dataset

REFERENCES

[1] R. Das, D.K. Bhattacharyya, and J.K. Kalita. Clustering gene expression data using an effective dissimilarity measure. accepted for publication in the special issue of International Journal of Computational Bioscience, 2009.

[2] D. Stekel. *Microarray Bioinformatics*. Cambridge University Press, Cambridge, UK., 2006.

[3] D. Jiang, C. Tang, and A. Zhang. Cluster analysis for gene expression data: A survey. Available: www.cse.buffalo.edu/DBGROUP/bioinformatics/ papers/survey.pdf., 2003.

[4] M. Ester, H. P. Kriegel, J. Sander, M. Wimmer, and X. Xu. An incremental clustering for mining in a data warehousing environment. In *Proceedings of the 24th VLDB Conference*, New York, USA, 1998.

[5] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of International Conference on Knowledge Discovery in Databases and Data Mining (KDD-96)*, pages 226–231, Portland, Oregon, 1996.

[6] R. Feldman, Y. Aumann, A. Amir, and H. Mannila. Efficient algorithms for discovering frequent sets in incremental databases. In *Proceedings of ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 59–66, Tucson, AZ, 1997.

[7] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 626–635, New York, NY, USA, 1997. ACM.

[8] L. Tao and S.A. Sarabjot. Hirel: An incremental clustering algorithm for relational datasets. In *Proceedings of Eighth IEEE International Conference on Data Mining*, pages 887–892, 2008.

[9] S. Asharaf, M. Narasimha, and S.K. Shevade. Rough set based incremental clustering of interval data. *Pattern Recognition Letters*, 27:515–519, 2006.

[10] Y. Lu, S. Lu, F. Fotouhi, Y. Deng, and S.J. Brown. Incremental genetic k-means algorithm and its application in gene expression data analysis. *BMC Bioinformatics*, 5(172), 2004.

[11] R. Ruiz, J.C. Riquelme, and J.S. Aguilar-Ruiz. Incremental wrapper-based gene selection from microarray data for cancer classification. *Pattern Recognition*, 39:23832392, 2006.

[12] R. Das, D.K. Bhattacharyya, and J.K. Kalita. A new approach for clustering gene expression time series data. *International Journal of Bioinformatics Reasearch and Applications*, 5(3):310–328, 2009.

[13] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: The order-preserving submatrix problem. In *Proc. Of the 6th Annual Int. Conf. on Computational Biology*, pages 49–57, New York, USA, 2002. ACM Press.

[14] J.L. DeRisi and P.O. Iyer, V.R.and Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278:680–686, 1997.

[15] R. J. Cho, M. Campbell, E. Winzeler, L Steinmetz, et al. A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol. Cell*, 2(1):6573, 1998.

[16] X. Wen, S. Fuhrman, G.S. Michaels, D.B. Carr, S. Smith, J.L. Barker, and R. Somogyi. Large-scale temporal gene expression mapping of central nervous system development. *PNAS*, 95(1):334–339, 1998.

[17] J.B. McQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symp. Math. Statistics and Probability*, volume 1, pages 281–297, 1967.

[18] M. Eisen, P. Spellman, P. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. In *Proceedings of National Academy of Sciences*, volume 95, pages 14863–14868, 1998.

[19] F. Gibbons and F. Roth. Judging the quality of gene expression based clustering methods using gene annotation. *Genome Research*, 12:1574–1581, 2002.